

1/5/1

DIALOG(R)File 347:JAPIO

(c) 2004 JPO & JAPIO. All rts. reserv.

04637390 \*\*Image available\*\*

PARALLEL PROCESSING SYSTEM

PUB. NO.: 06-309290 [ JP 6309290 A]

PUBLISHED: November 04, 1994 (19941104)

INVENTOR(s): ANZAI FUMIHIKO

APPLICANT(s): FUJI ELECTRIC CO LTD [000523] (A Japanese Company or Corporation), JP (Japan)  
FUJI FACOM CORP [470926] (A Japanese Company or Corporation), JP (Japan)

APPL. NO.: 05-097424 [JP 9397424]

FILED: April 23, 1993 (19930423)

INTL CLASS: [5] G06F-015/16; G06F-009/46; G06F-009/38

JAPIO CLASS: 45.4 (INFORMATION PROCESSING -- Computer Applications); 45.1 (INFORMATION PROCESSING -- Arithmetic Sequence Units)

JAPIO KEYWORD:R131 (INFORMATION PROCESSING -- Microcomputers & Microprocessors)

#### ABSTRACT

PURPOSE: To validly use each processor in the parallel processing system of a multiprocessor system.

CONSTITUTION: Peculiar identifiers are assigned to two schedulers 21-1 and 21-2, and those schedulers operate the scheduling of program modules to be executed according to each individual sequence description to a shared EXQ 31. Four job processors 41-1-41-4 competitively extract a program module number and the identifier of the scheduler in which the program module number is written from an EXQ 31, and execute the program module corresponding to the number. Then, after the execution is ended, the program module number is written in an EDQ corresponding to the identifier. The schedulers extract the number of the program module whose execution is ended from the private EDQ, and recognize the end of the execution.

(19)日本国特許庁(JP)

(12) 公開特許公報(A)

(11)特許出願公開番号

特開平6-309290

(43)公開日 平成6年(1994)11月4日

(51)Int.Cl. <sup>5</sup>	識別記号	庁内整理番号	F I	技術表示箇所
G 0 6 F 15/16	4 3 0	9190-5L		
9/46	3 6 0 B	8120-5B		
// G 0 6 F 9/38	3 7 0 X	9193-5B		

審査請求 未請求 請求項の数 3 O L (全 12 頁)

(21)出願番号 特願平5-97424

(22)出願日 平成5年(1993)4月23日

(71)出願人 000005234

富士電機株式会社

神奈川県川崎市川崎区田辺新田1番1号

(71)出願人 000237156

富士ファコム制御株式会社

東京都日野市富士町1番地

(72)発明者 安西 文彦

東京都日野市富士町1番地 富士ファコム  
制御株式会社内

(74)代理人 弁理士 大菅 義之

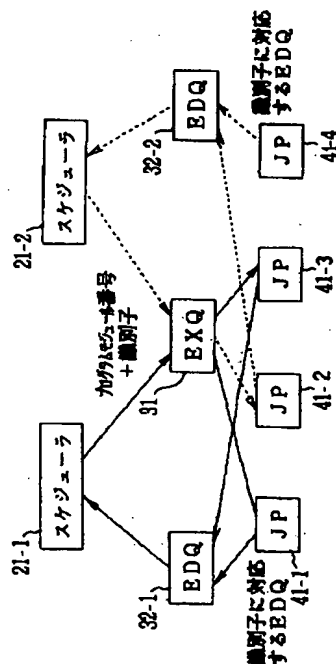
(54)【発明の名称】 並列処理方式

(57)【要約】

【目的】 本発明はマルチプロセッサシステムにおける並列処理方式に関し、各プロセッサの有効利用を図ることを目的とする。

【構成】 2台のスケジューラ21-1、21-2にはそれぞれ固有の識別子が割り当てられ、これらのスケジューラはそれぞれ個別の順序記述に従って実行可能になったプログラムモジュールを共用のEXQ31にスケジューリングする。4台のジョブ・プロセッサ41-1～41-4はEXQ31から競合してプログラムモジュール番号とこれを書き込んだスケジューラの識別子を取り出し、この番号に対応するプログラムモジュールを実行する。そして、この実行終了後、上記識別子に対応するEDQに上記プログラムモジュール番号を書き込む。スケジューラは専用のEDQから実行の終了したプログラムモジュールの番号を取り出し、その終了を知る。

本発明の第1実施例のシステム構成図



## 【特許請求の範囲】

【請求項1】 複数の処理単位を内蔵している複数のプロセッサ(1-1), (1-2), ... (1-N)

と、

実行可能な処理単位の待ち行列である実行可能待ち行列(2)と、

前記複数の処理単位の実行順序が記述された順序記述に従って実行可能になった処理単位を前記実行可能待ち行列(2)に入れる複数のスケジューラ(3-1), (3-2), ... (3-M)と、

該各スケジューラ(3-1), (3-2), ... (3-M)それぞれに対し1対1に対応して設けられた実行が終了した処理単位の待ち行列である複数の実行終了待ち行列(4-1), (4-2), ... (4-M)とを備え、

前記複数のスケジューラ(3-1), (3-2), ... (3-M)はそれぞれに割り当てられた個別の順序記述に従って実行可能になった処理単位を順次前記実行可能待ち行列(2)に入れ、前記複数のプロセッサ(1-1), (1-2), ... (1-N)は前記実行可能待ち行列(2)から競合して実行可能状態にある処理単位を取り出して実行し、実行終了後この処理単位をこれをスケジューリングしたスケジューラ(3-i; i=1, 2, ... M)に対応する実行終了待ち行列(4-i; i=1, 2, ... M)に入れる、  
ことを特徴とする並列処理方式。

【請求項2】 複数の処理単位を内蔵している複数のプロセッサ(5-1), (5-2), ... (5-K)

と、

前記複数の処理単位の実行順序が記述された順序記述に従って実行可能になった処理単位をスケジューリングするスケジューラ(6-1), (6-2), ... (6-L)と、

上記各スケジューラ(6-1), (6-2), ... (6-L)それぞれに対し1対1に対応して設けられた実行可能な処理単位の待ち行列である優先順位が付けられた複数の実行可能待ち行列(7-1), (7-2), ... (7-L)と、

前記複数のスケジューラ(6-1), (6-2), ... (6-L)それぞれに対し1対1に対応して設けられた実行が終了した処理単位の待ち行列である複数の実行終了待ち行列(8-1), (8-2), ... (8-L)とを備え、

前記複数のスケジューラ(6-1), (6-2), ... (6-L)はそれぞれに割り当てられた個別の順序記述に従って実行可能になった処理単位を対応する実行可能待ち行列(7-1), (7-2), ... (7-L)に入れ、前記複数のプロセッサ(5-1), (5-2), ... (5-K)は現在空き状態となっていない実行可能待ち行列(7-1), (7-2), ... (7-

-L)の中で最も優先順位の高い実行可能待ち行列(7-j; j=1, 2, ... L)から優先して実行可能な処理単位を取り出して実行し、この処理単位の実行後この処理単位をこれをスケジューリングしたスケジューラ(6-j; j=1, 2, ... L)に対応する実行終了待ち行列(8-j; j=1, 2, ... L)に入れる、  
ことを特徴とする並列処理方式。

【請求項3】 各実行可能待ち行列(7-1), (7-2), ... (7-L)から空き状態にあるか否かを示す情報を入力し、空き状態にない最も優先順位の高い実行可能待ち行列(7-j; j=1, 2, ... L)の先頭にある処理単位を出力する選択手段(9)をさらに備え、

前記複数のプロセッサ(5-1), (5-2), ... (5-K)は該選択手段(9)を介し実行すべき処理単位を取り出す、  
ことを特徴とする請求項2記載の並列処理方式。

## 【発明の詳細な説明】

## 【0001】

【産業上の利用分野】本発明は、マルチプロセッサシステムにおける並列処理方式に関し、特に複数の順序記述に従って複数の処理単位をスケジューリングする方式に関する。

【0002】尚、ここで処理単位とは、例えばタスク(プロセス)、スレッド、サブルーチン、ステートメント、関数などのデータ処理の単位であり、以後の説明では、プログラムモジュールと呼称する。また、順序記述とはタスクグラフ等のように1つのジョブを構成する複数の処理単位の実行順序が記述されたものであり、並行して実行可能な処理単位についても記述されている。

## 【0003】

【従来の技術】シングルプロセッサ方式では高速化に限界があることが明らかになってきたことにより、マルチプロセッサ方式による並列処理に対するニーズが高まっている。このマルチプロセッサ方式はMIMD(Multi Instruction Stream MultipleData Stream)方式の一つであり、複数のプロセッサ全体であるまとまった処理を実行する。

【0004】このマルチプロセッサ方式の目的の一つとして、同一処理を複数のプロセッサで分担(負荷分散)して高速化を図ることがあげられる。ここで、従来の並列処理方式を図8に示す。

【0005】同図に示すように、従来は1台のスケジューラ11、複数台のジョブ・プロセッサ(JP)21-1, 21-2, 21-3, ... 21-N, 1つの実行キュー(EXQ)31, 1つの終了キュー(EDQ)32, 及び不図示の共有データメモリから構成されていた。

【0006】上記構成において、スケジューラ11は、順序記述に従って実行すべきプログラムモジュールの番

## 3

号をFIFO (First in First Out) メモリから成るEXQ (Execution Que) 31に書き込む。

【0007】複数のジョブ・プロセッサ21-1, 21-2, 21-3, ... 21-Nは、マイクロプロセッサ並びにローカルメモリ等から成り、このローカルメモリ内に複数のプログラムモジュールを格納している。このプログラムモジュールは、ジョブ・プロセッサ21-i ( $i=1, 2, 3, \dots, N$ ) が実行可能な言語であればいかなる言語で記述されていてもよい。

【0008】EXQ 31は、実行可能状態にあるプログラムモジュールの番号の待ち行列(キュー)であり、データを先入れ先出し(FIFO)するFIFOメモリから成る。

【0009】EDQ (END Que) 32も同様にFIFOメモリから成り、実行が終了したプログラムモジュールが各ジョブ・プロセッサ21-1, 21-2, 21-3, ... 21-Nにより書き込まれる。

【0010】このような構成において、スケジューラ11は順序記述に従って実行可能になったプログラムモジュールの番号を順にEXQ 31に書き込む。すなわち、あるプログラムモジュール(先行プログラムモジュール)の実行が終了すると、順序記述から次に実行すべきプログラムモジュール(後続プログラムモジュール)を取り出しこのプログラムモジュールの番号をEXQ 31に書き込む処理を順序記述に従って行う。この場合、同時に並列(並行)して実行可能なプログラムモジュールが複数あった場合には、これらのプログラムモジュールの番号をEXQ 31に続けて書き込む。

【0011】複数のジョブ・プロセッサ21-1, 21-2, 21-3, ... 21-Nは、EXQ 31からプログラムモジュール番号を競合しながら取り出し、この番号を有するプログラムモジュールを実行する。そして、実行を終了したならば、このプログラムモジュールの番号をEDQ 32に書き込む。

【0012】スケジューラ11は実行が終了したプログラムモジュールの番号をEDQ 32から取り出し、このことにより実行終了状態に遷移したプログラムモジュールを知る。

【0013】以上において、スケジューラ11と複数のジョブ・プロセッサ21-1, 21-2, 21-3, ... 21-Nは、EXQ 31及びEDQ 32をインタフェースとして互いに独立に動作する。すなわち、スケジューラ11は与えられた順序記述に従いながら、並列(並行)して同時に実行可能なプログラムモジュールの番号をEXQ 31に書き込んでいく。一方、各ジョブ・プロセッサ21-1, 21-2, 21-3, ... 21-NはEXQ 31からプログラムモジュール番号を取り出し、この番号に対応するプログラムモジュールを実行する。このとき、複数台のジョブ・プロセッサ21-1, 21-2, 21-3, ... 21-Nは互いに独立

## 4

にプログラムモジュール番号をEXQ 31から取り出し、独立にこれらの番号を有するプログラムモジュールを実行する。従って、最大、ジョブ・プロセッサ21-i ( $i=1, 2, 3, \dots, N$ ) の台数に等しい個数のプログラムモジュールが同時に並列して実行可能である。

【0014】

【発明が解決しようとする課題】 上述したように、従来の並列処理方式では1台のスケジューラ11のみが1つの順序記述に従ってEXQ 21にプログラム・モジュールのスケジューリングを行っていた。一般に順序記述に従ってプログラム・モジュールのスケジューリングを行った場合、並列実行可能なプログラム・モジュールの数は時間的に変動する場合が多い。このため、原理上では、常時ジョブ・プロセッサ21-i ( $i=1, 2, \dots, N$ ) の台数Nに等しい個数だけプログラム・モジュールを同時に並列して実行可能であるにもかかわらず、休眠状態となるジョブ・プロセッサ21-i ( $i=1, 2, \dots, N$ ) の数が増加してしまうときがあり、ジョブ・プロセッサ21-i ( $i=1, 2, \dots, N$ ) を有効に利用しているとは言い難かった。

【0015】本発明は、プロセッサ(上記ジョブ・プロセッサに対応)が休眠状態になる頻度を少なくして、プロセッサの利用率を高め複数のプロセッサを有効活用できるようにすることを目的とする。また、さらにはこの目的を複数の実行順序を優先順位を付けながら実行していくシステムにおいても実現可能にするを第2の目的とする。

【0016】

【課題を解決するための手段】 図1は請求項1記載の第1の発明の原理ブロック図である。この第1の発明は、複数の処理単位を内蔵している複数のプロセッサ1-1, 1-2, ... 1-Nと、実行可能な処理単位の待ち行列である実行可能待ち行列2と、前記複数の処理単位の実行順序が記述された順序記述に従って実行可能になった処理単位を前記実行可能待ち行列2に入れる複数のスケジューラ3-1, 3-2, ... 3-Mと、該複数のスケジューラ3-1, 3-2, ... 3-Mそれぞれに対し1対1に対応して設けられた実行が終了した処理単位の待ち行列である複数の実行終了待ち行列4-1, 4-2, ... 4-Mとを備え、前記複数のスケジューラ3-1, 3-2, ... 3-Mはそれぞれに割り当てられた個別の順序記述に従って実行可能になった処理単位を順次前記実行可能待ち行列2に入れ、前記複数のプロセッサ1-1, 1-2, ... 1-Nは前記実行可能待ち行列2から競合して実行可能状態にある処理単位を取り出して実行し、実行終了後この処理単位をこれをスケジューリングしたスケジューラ3-i;  $i=1, 2, \dots, M$ に対応する実行終了待ち行列4-i;  $i=1, 2, \dots, M$ に入れる。

## 5

【0017】次に、図2は請求項2記載の第2の発明の原理ブロック図である。この第2の発明は、複数の処理単位を内蔵している複数のプロセッサ5-1, 5-2, ……5-Kと、前記複数の処理単位の実行順序が記述された順序記述に従って実行可能になった処理単位をスケジューリングするスケジューラ6-1, 6-2, ……6-Lと、上記各スケジューラ6-1, 6-2, ……6-Lそれぞれに対し1対1に対応して設けられた実行可能な処理単位の待ち行列である優先順位が付けられた複数の実行可能待ち行列7-1, 7-2, ……7-Lと、前記複数のスケジューラ6-1, 6-2, ……6-Lそれぞれに対し1対1に対応して設けられた実行が終了した処理単位の待ち行列である複数の実行終了待ち行列8-1, 8-2, ……8-Lとを備え、前記複数のスケジューラ6-1, 6-2, ……6-Lはそれぞれに割り当てられた個別の順序記述に従って実行可能になった処理単位を対応する実行可能待ち行列7-1, 7-2, ……7-Lに入れ、前記複数のプロセッサ5-1, 5-2, ……5-Kは現在空き状態となっていない実行可能待ち行列7-1, 7-2, ……7-Lの中で最も優先順位の高い実行可能待ち行列7-j; j=1, 2, ……Lから優先して実行可能な処理単位を取り出して実行し、この処理単位の実行終了後この処理単位をスケジューリングしたスケジューラ6-j; j=1, 2, ……Lに対応する実行終了待ち行列8-j; j=1, 2, ……Lに入れる。

【0018】この第2の発明において、例えば請求項3記載のように(図3参照)、各実行可能待ち行列7-1, 7-2, ……7-Lから空き状態にあるか否かを示す情報を入力し、空き状態にない最も優先順位の高い実行可能待ち行列7-j; j=1, 2, ……Lの先頭にある処理単位を出力する選択手段9をさらに備え、前記複数のプロセッサ5-1, 5-2, ……5-Kは該選択手段9を介し実行すべき処理単位を取り出すような構成にしてもよい。

## 【0019】

【作用】上記第1の発明においては、各スケジューラ3-1, 3-2, ……3-Mはそれぞれ個別の順序記述に従って、実行可能になった処理単位を実行可能待ち行列2に入れるスケジューリング処理を行う。

【0020】そして、複数のプロセッサ1-1, 1-2, ……1-Nは、実行可能待ち行列2から競合して処理単位を取り出しこれを実行する。そして、実行を終了すると、この処理単位をスケジューリングしたスケジューラ3-i (i=1, 2, ……M)に対応する実行終了待ち行列4-i (i=1, 2, ……M)に入れる。

【0021】このように、各スケジューラ3-1, 3-2, ……3-M毎に専用の実行終了待ち行列4-1, 4-2, ……4-Mを設け、各スケジューラ3-1,

## 6

3-2, ……3-Mがそれぞれ個別の順序記述に従って複数のプロセッサ1-1, 1-2, ……1-Nによって共用される実行可能待ち行列2に処理単位をスケジューリングしていくようにしたので、システム内で同時に複数の順序記述に従って処理単位をスケジューリングできる。そして、このように複数の順序記述に従ってスケジューリングされる処理単位を複数のプロセッサ1-1, 1-2, ……1-Nが共用して実行する。従って、従来のように1つの順序記述に従って処理単位をスケジューリングしていく場合に比べ各プロセッサが休眠状態となる頻度は減少し、複数のプロセッサ1-1, 1-2, ……1-Nを従来よりも有効活用できるようになる。

【0022】また、第2の発明においては、上記第1の発明と同様に各スケジューラ6-1, 6-2, ……6-Lに対し専用の実行終了待ち行列8-1, 8-2, ……8-Lを設けるのに加え、さらに専用の実行可能待ち行列7-1, 7-2, ……7-Lも各スケジューラ6-1, 6-2, ……6-Lに対し設ける。また、これらの実行可能待ち行列7-1, 7-2, ……7-Lに対し優先順位を割り当てる。

【0023】そして、各スケジューラ6-1, 6-2, ……6-Lは個別の順序記述に従って各々が専用する実行可能待ち行列7-1, 7-2, ……7-Lに処理単位をスケジューリングする。複数のプロセッサ5-1, 5-2, ……5-Kはこれらのスケジューリングされた処理単位を、現在空き状態となっていない実行可能待ち行列7-1, 7-2, ……7-Lの中で最も優先順位の高い実行可能待ち行列7-j (j=1, 2, ……L)から優先的に取り出して実行する。そして、実行が終了するとこの処理単位をこれをスケジューリングしたスケジューラ6-j (j=1, 2, ……L)が専用する実行終了待ち行列8-j (j=1, 2, ……L)に入れる。

【0024】このように、この第2の発明においては優先順位の高い実行可能待ち行列内にある処理単位が優先順位の低い実行可能待ち行列内にある処理単位よりも優先して実行される。上述したように、各実行可能待ち行列7-1, 7-2, ……7-Lにはそれぞれ個別の順序記述に従って処理単位がスケジューリングされていく。このため、複数の順序記述は優先順位が付与されて実行される。

【0025】したがって、上記第1の発明と同様に複数のプロセッサを従来よりも有効に活用できると共に、これを複数の実行順序を優先順位に従って優先処理しながら実現できる。

## 【0026】

【実施例】以下、図面を参照しながら本発明の実施例を説明する。図4は、本発明の第1実施例である並列処理装置のシステム構成図である。

【0027】同図に示すように、この並列処理装置においては、2台のスケジューラ21-1、21-2が設けられている。そして、これらのスケジューラ21-1、21-2が共通に使用する実行可能待ち行列であるEXQ (Execute Que) 31が1つ設けられ、さらに各スケジューラ21-1、21-2がそれぞれ専用して使用する実行終了待ち行列である2つのEDQ (END Que) 32-1、32-2が設けられている。

【0028】また、互いに独立して個別のプログラムモジュールを実行可能な4台のジョブ・プロセッサ(JP) 41-1、41-2、41-3、41-4が設けられている。

【0029】さらに、特に図示してはいないが、これらのジョブ・プロセッサ41-1、41-2、41-3、41-4間で共有データを授受するために使用される共有データメモリが、各ジョブ・プロセッサ41-1、41-2、41-3、41-4とバスで接続されている。

【0030】これら各ジョブ・プロセッサ41-j (j=1, 2, 3, 4)は、マイクロプロセッサ、及びこのマイクロプロセッサにローカルバスで接続されたローカルメモリ等から成り、このローカルメモリに処理単位に相当するプログラムモジュールや非共有データを格納している。

【0031】また、EXQ 31及びEDQ 32-1、32-2は共に先入れ先出し方式のFIFOメモリ (Fast In Fast Out Memory)から成る。各スケジューラ21-1、21-2は、個別の識別子(スケジューラ識別子)が割り当てられており、それぞれに与えられた個別の順序記述に従って実行すべきプログラムモジュールのスケジューリング、すなわち実行すべきプログラムモジュールの番号(プログラムモジュール番号)のEXQ 31への書き込みを行う。尚、このプログラムモジュール番号の書き込みの際には自己の識別子も書き込む。また専用のEDQ 32-1またはEDQ 32-2から実行が終了したプログラムモジュールの番号を取り出す。

【0032】次に、上記構成の第1実施例の動作を説明する。各スケジューラ21-1、21-2は、それぞれに与えられた順序記述を基に実行可能状態になったプログラムモジュールの番号を自己の識別子と共に順次EXQ 31に書き込んでいく。

【0033】複数のジョブ・プロセッサ41-1、41-2、41-3、41-4は、プログラムモジュールの実行が可能な状態になると、EXQ 31からプログラムモジュールの番号とこれに対応して書き込まれているスケジューラの識別子とを互いに競合しながら取り出し、このプログラムモジュール番号に対応するプログラムモジュールを実行する。この場合、各ジョブ・プロセッサ41-j (j=1, 2, 3, 4)はEXQ 31に書き込まれた順にプログラムモジュール番号を取り出すので、各プロセッサ41-j (j=1, 2, 3, 4)が取り出

すプログラムモジュールはスケジューラ21-1またはスケジューラ21-2の内どちらによってスケジューリングされたものであるか特定されない。

【0034】そして、各ジョブ・プロセッサ41-j (j=1, 2, 3, 4)はプログラムモジュールの実行を終了すると、このプログラムモジュールの番号を、この番号と共にEXQ 31から取り出した識別子を有するスケジューラ21-1またはスケジューラ21-2に対応するEDQ 32-1またはEDQ 32-2に書き込む。すなわち、実行が終了したプログラムモジュールの番号はこのプログラムモジュールをスケジューリングしたスケジューラ21-1またはスケジューラ21-2専用のEDQ 32-1またはEDQ 32-2に書き込まれることになる。

【0035】各スケジューラ21-1、21-2はそれぞれ自己の識別子に対応するEDQ 32-1、32-2から自己がスケジューリングしたプログラムモジュールの番号を取り出し、そのプログラムモジュールの状態遷移、すなわち実行中から実行終了への移行を知る。

【0036】ここで、各ジョブ・プロセッサ内のEDQ選択回路の一例を図5に示す。このEDQ選択回路はEXQ 31から取り出すスケジューラ21-1またはスケジューラ21-2の識別子に従って実行が終了したプログラムモジュールの番号を当該EDQ 32-1またはEDQ 32-2に書き込む。

【0037】同図において、識別子レジスタ411、番号レジスタ412は、それぞれEXQ 31から取り出されるスケジューラの識別子及びプログラムモジュールの番号を格納するレジスタである。

【0038】セレクト413は、上記番号レジスタ412からプログラムモジュールの番号をデータとして入力し、識別子レジスタ411からスケジューラの識別子を選択信号として入力する。そして、この選択信号がスケジューラ21-1の識別子であればEDQ 32-1へ、一方スケジューラ21-2の識別子であればEDQ 32-2へ、上記入力されるプログラムモジュールの番号を選択出力する。

【0039】尚、番号レジスタ412に格納されるプログラムモジュールの番号はプロセッサ414により読み出され、プロセッサ414はこの番号を有するプログラムモジュールを不図示のローカルメモリから読み出して実行する。そして、プロセッサ414はこのプログラムモジュールの実行を終了すると上記セレクト413から選択出力されているプログラムモジュール番号を当該EDQ (32-1または32-2)へ書き込む。

【0040】このようにこの第1実施例においてはスケジューラ→EXQ→ジョブ・プロセッサ→EDQのループすなわちプログラムモジュールの実行環境が、図4中においてそれぞれ実線と破線の矢印で示したように2組存在することになる(但し、EXQ 31とジョブ・プロ

セッサ41-1~41-4は共通)。従って、本実施例では2組の順序記述について同時に並列してスケジューリング処理できる。また、上記ループの個数はスケジューラとEDQの個数によって決定されるため(同一数となる)、スケジューラとEDQの個数を増加することにより同時に並列してスケジューリングできる順序記述の数をさらに増加できる。

【0041】このように第1実施例においては、それぞれ個別の順序記述に従ってプログラムモジュールのスケジューリングを行う複数のスケジューラがシステム内の全てのジョブ・プロセッサに実行すべきプログラムモジュールを供給するので、ジョブ・プロセッサが休眠状態となる頻度は減少する。したがって、ジョブ・プロセッサの稼働率は高まり、ジョブ・プロセッサは有効活用される。

【0042】次に、図6は本発明の第2実施例である並列処理装置のシステム構成図である。同図に示す2台のスケジューラ121-1、121-2に対しては、上述した第1実施例と同様にそれぞれ専用のEDQ132-1、132-2が設けられている。ここで、上述した第1実施例との構成上の違いは、2つのEXQ131-1、131-2がそれぞれ上記スケジューラ121-1、121-2専用に設けられていることである。

【0043】これらEXQ131-1、131-2、及びEDQ132-1、132-2は、いずれもFIFOメモリである。また、EXQ131-1、131-2には優先順位が割り当てられており、EXQ131-1の方がEXQ131-2よりも優先順位が高くなっている。

【0044】セクタ151は優先順位の高いEXQ131-1にプログラムモジュール番号が有る場合には、EXQ131-2にプログラムモジュール番号が格納されているのかかわらず、EXQ131-1に格納されているプログラムモジュール番号の方を優先して対応するスケジューラ(121-1)の識別子と共に選択出力する。

【0045】4台のジョブ・プロセッサ141-1~141-4は上述した第1実施例の4台のジョブ・プロセッサと同一の構成となっており、上記セクタ151から出力されるプログラムモジュール番号と、これに対応して書き込まれているスケジューラの識別子とを互いに競合しながら取り込む。そして、この番号に対応するプログラムモジュールを実行し、実行が終了するとこの番号を取り込んでいた識別子を有するスケジューラ(121-1または121-2)に対応するEDQ(122-1または122-2)に書き込む。

【0046】各スケジューラ121-1、121-2は、それぞれ専用のEDQ132-1、132-2からプログラムモジュール番号を取り出し、この番号を有するプログラムモジュールが実行中から実行終了の状態に

遷移したことを知る。そして、順序記述から次に実行可能なプログラムモジュール(並列実行可能なプログラムモジュール)を全て求め、これらのプログラムモジュールの番号を、それぞれ対応するEXQ131-1、131-2に書き込む。

【0047】ここで、この第2実施例の特徴部分である優先順位の高いEXQ131-1の方から優先的にプログラムモジュール番号を取り出す回路を図7に示す。各EXQ131-1、131-2はそれぞれエンプティフラグ131a、132aを備え、プログラムモジュール番号を1個も格納していないときにはこのフラグを立てる。これら各EXQ131-1、131-2のエンプティフラグ131a、132aの出力は共に上記セクタ151に選択信号として入力される。

【0048】また、このセクタ151には、2つのEXQ131-1、131-2からそれぞれの待ち行列の先頭の組データ(プログラムモジュール番号、スケジューラ識別子)がデータ入力される。

【0049】セクタ151は優先順位の高いEXQ131-1のエンプティフラグ131aが立っているとき以外は、常にEXQ131-1から入力される組データ(プログラムモジュール番号、スケジューラ識別子)の方を優先順位の低いEXQ131-2から入力される組データ(プログラムモジュール番号、スケジューラ識別子)よりも優先して出力する。そして、EXQ131-1のエンプティフラグ131aが立っておりかつEXQ131-2のエンプティフラグ131bが立っていないときにのみEXQ131-2から入力される上記組みデータを出力する。

【0050】このように、第2実施例においても上述した第1実施例と同様にプログラムモジュールの実行環境ともいうべきスケジューラ→EXQ→ジョブ・プロセッサ(共用)→EDQのループが二組存在する。従って、上記第1実施例と同様に2つの順序記述に従ってプログラムモジュールのスケジューリングが可能であり、これによりスケジューリングされたプログラムモジュールを4台のジョブ・プロセッサ141-1~141-4が並列して処理できる。

【0051】またこのスケジューリングにおいては上述したようにEXQ131-1の待ち行列にリスタンピングされているプログラムモジュールの方が優先され、EXQ131-1にはスケジューラ121-1が自己に割り当てられた順序記述に従ってプログラムモジュールのスケジューリングを行うことから、2つの順序記述の内スケジューラ121-1に割り当てられた順序記述の方がスケジューラ121-2に割り当てられた順序記述よりも優先的に処理されることになる。従って、第2実施例においては2つの順序記述を優先順位を付けながら実行することが可能である。

【0052】尚、この第2実施例においても各ジョブ・

プロセッサ141-1~141-4における実行の終了したプログラムモジュールの番号を当該EDQ(132-1または132-2)へ書き込む機能は上述した上記第1実施例と同様に図5に示す回路により実現できる。

【0053】

【発明の効果】請求項1記載の発明(第1の発明)においては、1つの順序記述を実行するスケジューラ→実行可能な待ち行列(共用)→プロセッサ(共用)→実行終了待ち行列のループが複数組存在するので、複数の順序記述が矛盾することなく複数のプロセッサによって実行される。換言すれば、複数のプロセッサが複数の順序記述に従ってスケジューリングされる処理単位を実行できる。従って、実行可能状態にある処理単位の個数の時間的変動が小さくなり、各プロセッサが休眠状態となる頻度が小さくなる。この結果、複数のプロセッサを従来よりも有効活用できるようになる。

【0054】また、請求項2記載の発明(第2の発明)においては、上記ループにおいて1つの順序記述毎に専用の実行可能待ち行列が割り当てられる。そして、これらの実行可能待ち行列には優先順位が付与され、複数のプロセッサは空き状態にない実行可能待ち行列の中で最

も優先順位の高い実行可能待ち行列から優先的に処理単位を取り出してこれを実行する。したがって、上記第1の発明の効果に加え、複数の順序記述を優先順位に基づきながら処理できる。

【図面の簡単な説明】

【図1】本発明の原理ブロック図(その1)である。

【図2】本発明の原理ブロック図(その2)である。

【図3】本発明の原理ブロック図(その3)である。

【図4】本発明の第1実施例のシステム構成図である。

【図5】第1実施例のジョブ・プロセッサ内のEDQ選択回路の一構成例を示す図である。

【図6】本発明の第2実施例のシステム構成図である。

【図7】第2実施例における優先順位によるEXQの選択回路の一構成図である。

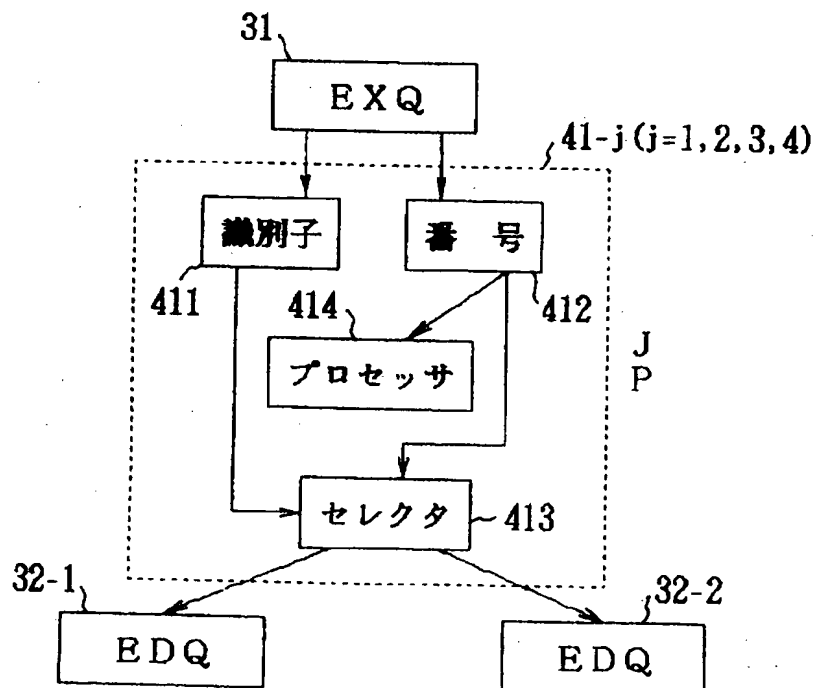
【図8】従来の並列処理方式を説明する図である。

【符号の説明】

1-1~1-N, 5-1~5-K	プロセッサ
2, 7-1~7-L	実行可能待ち行列
3-1~3-M, 6-1~6-L	スケジューラ
4-1~4-M, 8-1~8-L	実行終了待ち行列
9	選択手段

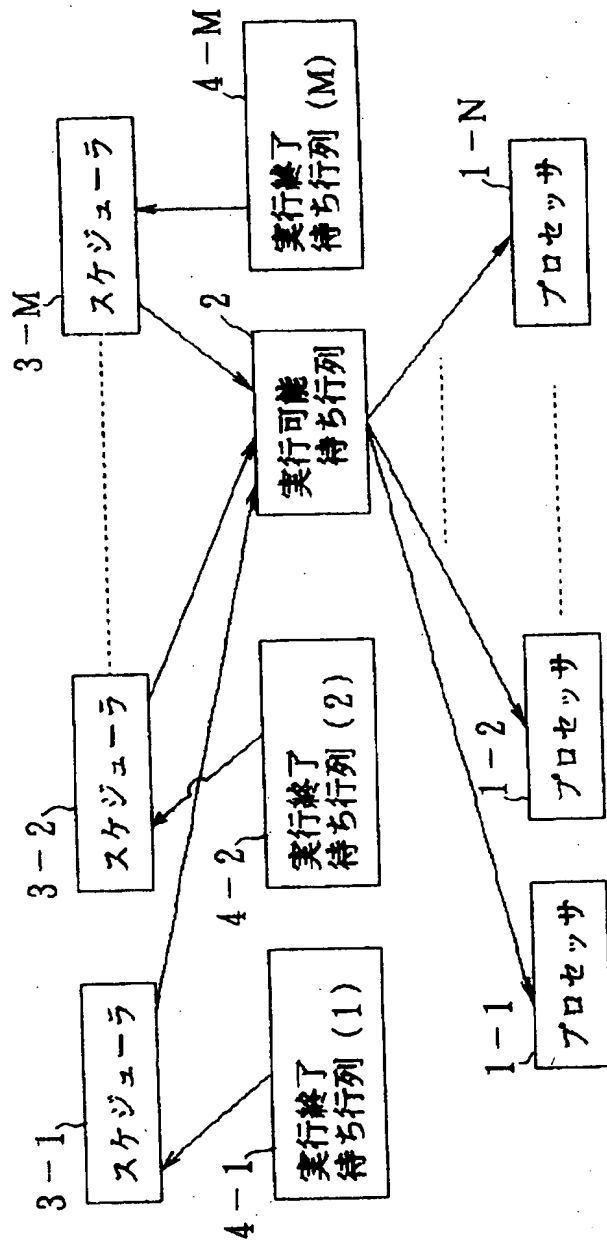
【図5】

第1実施例のジョブ・プロセッサ内のEDQ選択回路の一構成例を示す図



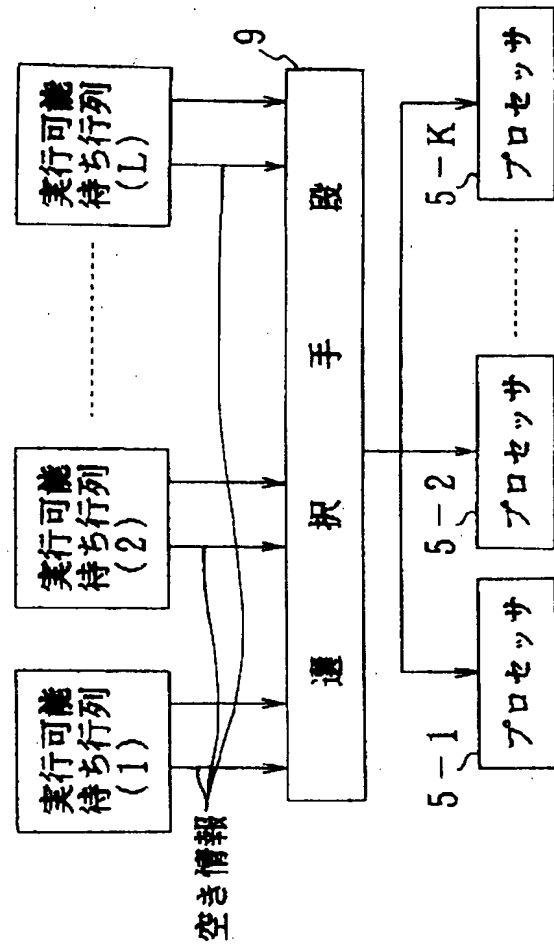
【図1】

本発明の原理ブロック図(その1)



【図3】

本発明の原理ブロック図(その3)

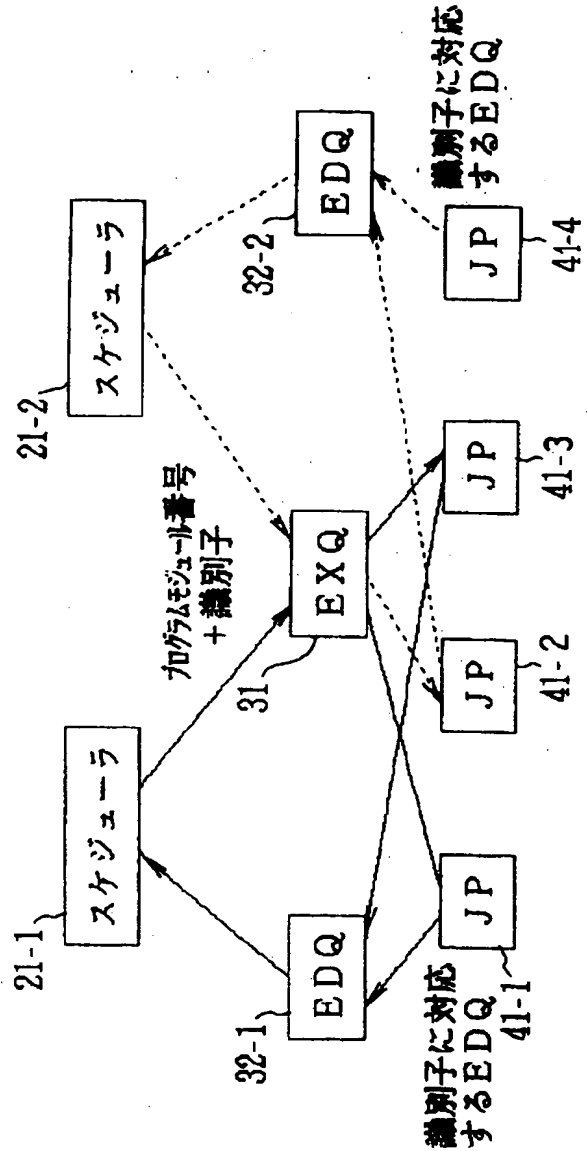
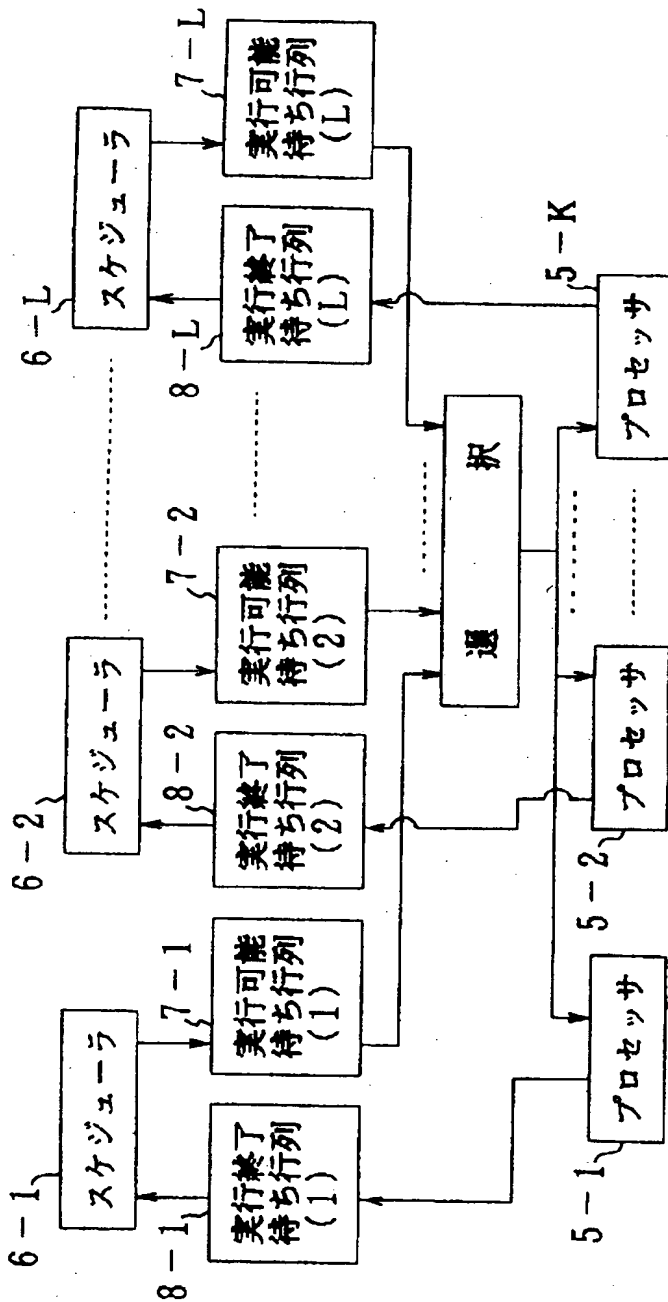


【図2】

【図4】

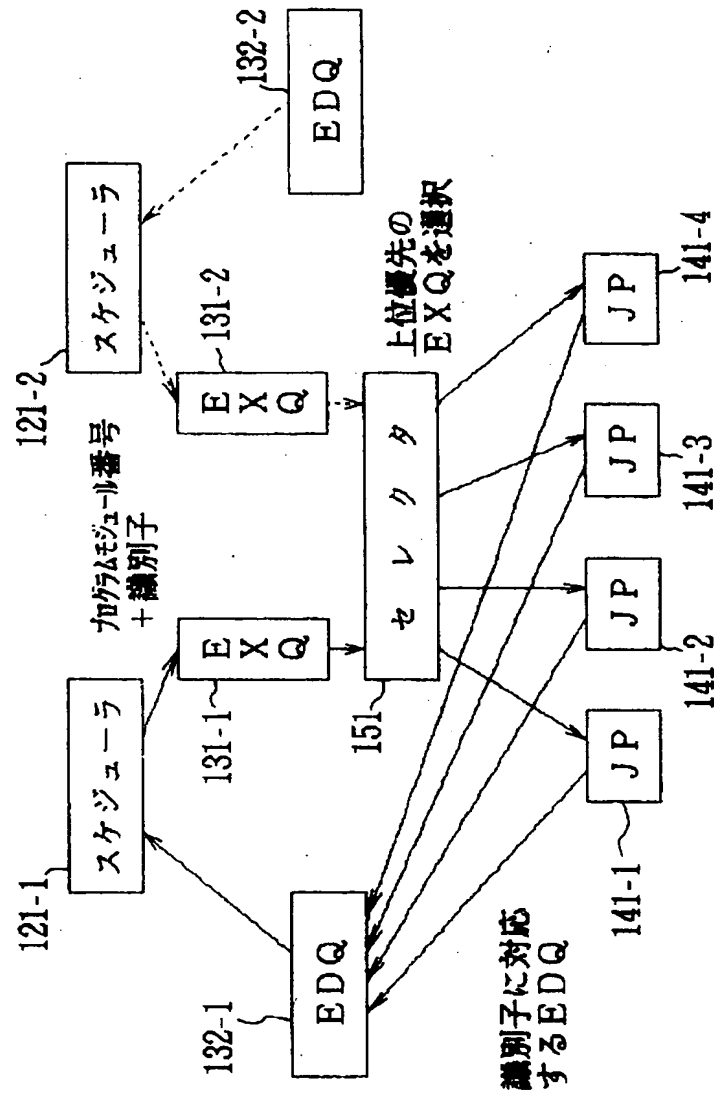
本発明の原理ブロック図(その2)

本発明の第1実施例のシステム構成図



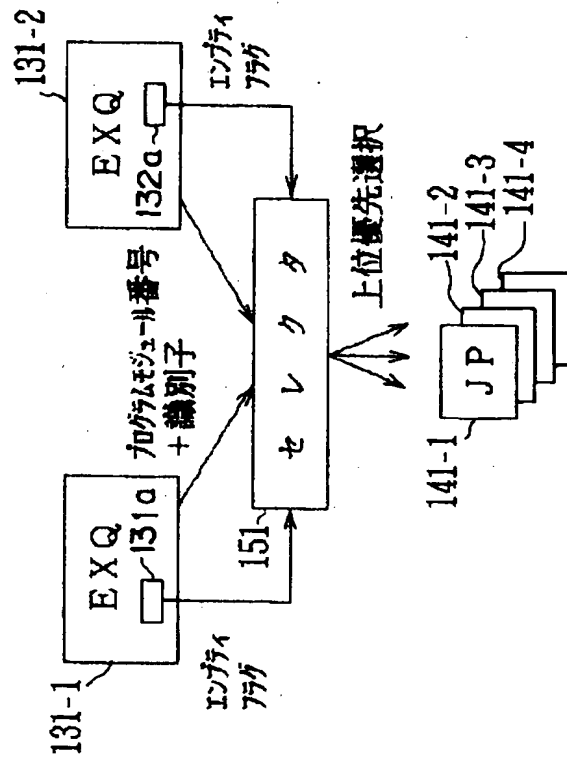
【図6】

## 本発明の第2実施例のシステム構成図



【図7】

第2実施例における優先順位による EXQ の  
選択回路の一構成図



【図8】

従来の並列処理方式を説明する図

